Remarks

Applicants thank the Examiner for the careful consideration given this Application. Reconsideration of this Application is now respectfully requested.

Upon entry of the foregoing Amendment, Claims 1-3, 8, 11-27, 29, and 32 are pending in the application, with Claims 15, 19, and 32 being the independent claims. Claims 4-7, 10, 20, 21, 28, 31, and 32 have been cancelled without prejudice to or disclaimer of the subject matter therein. New Claim 32 has been added.

Based on the above Amendment and the following Remarks, Applicants respectfully request that the Examiner reconsider all outstanding objections and rejections and that they be withdrawn.

Objection to the Specification

At Page 2, the Office Action objects to the specification based on a number of informalities and based on a need to better identify trademarks used in the specification. Applicants have reviewed the specification and have made appropriate changes to address these points, except for those cited in connection with Figs. 5 and 6, as well as some additional minor informalities. No new matter has been added.

In the case of the informalities cited in connection with Figs. 5 and 6, new versions of these drawings are being submitted concurrently with this Amendment and Reply. These new versions have been prepared in accordance with the drawings as originally filed, so it is

respectfully submitted that no new matter is added by these revised drawings. It is further respectfully submitted that the revised versions of Figs. 5 and 6 obviate the objections relating to Figs. 5 and 6.

Objection to the Drawings

At Page 2, the Office Action objects to the drawings on the basis of informalities in Figs. 1A, 4B, and 4C. Applicants are submitting concurrently with this Amendment and Reply revised versions of these drawings, which address the cited informalities. It is respectfully submitted that no new matter is added by the drawing amendments. Approval of the amended drawings, as well as of the substitute formal drawings submitted with this Amendment and Reply, including revised versions of Figs. 5 and 6 (discussed above), is respectfully requested.

Rejections under 35 U.S.C. § 112

At Page 3, the Office Action rejects Claim 1 under 35 U.S.C. § 112, first paragraph, as not being enabled. In particular, the Office Action alleges that the condition recited in step (d) of Claim 1 is applied to at least one of the attributes of the container and at least one of the attributes of the objects is not supported. The Office Action maintains that the disclosure only supports the case where the condition is applied to all the attributes of the container and the object, and in this connection, the Office Action refers to p. 21, line 3 to p. 22, line 7.

Applicants respectfully traverse this rejection. Attention is drawn to page 16, lines 21-27, where reference is made to a specific embodiment in which a bookmark (being one example of an

object) is displayed in a folder (being one example of container) if the bookmark and the folder share a common attribute (lines 27-28). This illustrates in a non-limiting manner an example where a condition is applied to only one attribute of the container and the object. The invention is not bound by this example, and in this connection, reference is also made to p. 20, line 28 – page 21, line 2, which provides support for the fact that the condition may be modified, depending upon the desired application.

For at least these reasons, Applicants respectfully submit that this rejection of Claim 1 should be withdrawn.

At Page 3 of the Office Action, Claim 1 is further rejected under 35 U.S.C. § 112, second paragraph, as being indefinite, citing insufficient antecedent basis for the word "set" in line 5 and that the meaning of "if a condition is met" should be clarified.

In response, Applicants have amended Claim 1 to explicitly recite, in lines 4 and 5, that the term "set" refers to a set of attributes. Applicants have also amended Claim 1 to clarify that the last clause in paragraph (d) modifies "condition." It is thus clear now that the condition is some condition applied to attributes of the container and of the object. Accordingly, Applicants respectfully submit that these rejections have been obviated and should be withdrawn.

At Page 4, the Office Action rejects Claims 2, 7, 19, and 24 under 35 U.S.C. § 112, second paragraph, as being indefinite for various reasons. Claim 7 has now been cancelled, rendering its rejection moot.

Claim 2 has amended in accordance with the proposal in the Office Action. Applicants would

like to comment that a container may be a new container or already existing one, since it is always possible to add a new container in accordance with the provisions of Step (b). The same applies to an object.

In connection with Claim 19, Applicants have amended "*vice versa*" in accordance with the suggested wording in the Office Action. Further in connection with Claim 19, the Office Action also asserts that Claim 19 is indefinite because "update" can not mean "adding" (i.e., a new attribute). Applicants respectfully traverse this basis for rejection. Attention is respectfully drawn to the *Dictionary of Computing* by IBM, specifically to the entry "update" on p. 604 (copy enclosed). The definition for "update" is given as "to **add**, change, or delete items" (emphasis added). Applicants, therefore, believe that the use of the term "update" in this sense in Claim 19 is clear.

Claim 24 was rejected on the basis of its inclusion of the wording, "like Microsoft™ File Explorer." In response, Applicants have substituted for this the language, "is displayed in a hierarchical, expandable and contractable manner," which describes the tree portion of the display in Microsoft™ File Explorer (i.e., the left panel of the display), and is thus supported by the claim as originally filed.

In view of the above, Applicants respectfully request that the rejections of Claims 2, 19, and 24 be withdrawn.

Rejections under 35 U.S.C. § 102

At Pages 4-6, the Office Action rejects Claims 1, 3, 5-7, 10, 13, 14, 28, 30, and 31 under 35 U.S.C. § 102(b) as being clearly anticipated by de Jucidibus (U.S. Patent No. 6,163,317). Applicants note that the issue date of de Judicibus (December 19, 2000) is subsequent to the filing date of this Application (March 9, 2000); hence, these rejections are being considered as rejections under 35 U.S.C. § 102(e). With regard to the rejections themselves, it is first noted that Claims 5-7, 10, 28, 30, and 31 have been cancelled, thus rendering moot their rejections. The rejections of Claims 1, 3, 13, and 14 are respectfully traversed for the following reasons.

Claim 1 has been amended to depend from Claim 15, which is respectfully submitted to be allowable over the cited prior art, as will be discussed below. Similarly, Claims 3, 13, and 14, depend from Claim 1 and thus from Claim 15. For at least these reasons, Claims 1, 3, 13, and 14 are allowable over the cited prior art.

Claim 1 is allowable over de Judicibus for the following further reasons (as are all of its dependent claims, Claims 2, 3, 8, 13, and 14). Claim 1 recites "providing a user interface for dynamically assigning attributes to said objects." The Office Action cites de Judicibus at col. 3, line 62 and col. 4, lines 38-45 as disclosing this limitation. Applicants respectfully submit, however, that this limitation is not taught in these portions, or in any other portions, of de Judicibus. In particular, **these portions of de Judicibus fail to provide for a user interface**, as claimed.

Claim 13 is allowable over de Judicibus for the following further reasons. Claim 13 recites that "at least one of said objects further contains object-related-data selectively displayed in said user interface." The Office Action cites Col. 4, lines 17-22 of de Judicibus as disclosing this feature. While even if this portion of de Judicibus does, *arguendo*, disclose the presence of some sort of object-related-data, **nowhere is it disclosed that such data is displayed in a user interface**, as claimed.

Claim 14 is allowable over de Judicibus for the following further reasons. Claim 14 recites, "applying at least one of the following to said set of attributes: adding at least one attribute, deleting at least one attribute and updating at least one attribute." It is noted that this refers to an operation of the **set of attributes**. In contrast, the cited portion of de Judicibus (Col. 4, lines 42-45) addresses whether the **attributes of a given object** are changed. That is, **there is no disclosure regarding a set of attributes from which the attributes of an object may be assigned, but merely disclosure saying that it is possible that the attributes of an object have changed**.

Rejections under 35 U.S.C. § 103

At Pages 6-7, the Office Action rejects Claims 2 and 8 under 35 U.S.C. § 103(a) as being unpatentable over de Jucidibus in view of Belove et al. (U.S. Patent No. 5,491,820). These rejections are respectfully traversed in view of the fact that Claims 2 and 8 depend from Claim 1, which, as discussed above, is respectfully submitted to be allowable over the cited prior art.

Furthermore, it is respectfully submitted that Belove et al. fails to remedy the deficiencies of de Judicibus.

At Pages 7-12, the Office Action rejects Claims 4, 9, 11, 12, 15-27, and 29 under 35 U.S.C. § 103(a) as being unpatentable over de Jucidibus in view of an article by Kamiya et al. ("Kamiya et al."). Claims 4, 9, 20, and 21 have been cancelled, thus rendering moot their rejections. The rejections of the remaining claims, Claims 11, 12, 15-19, 22-27, and 29, are now respectfully traversed for the following reasons.

All of the rejected claims here depend, directly or indirectly, from Claim 15 (note that Claims 11 and 12 have been amended to depend from Claim 15). Claim 15 will, therefore, be addressed first.

Claim 15 recites a method for sharing objects among a community of users. Step (a) recites, "providing a user replica that includes objects that are assigned, each, with at least one attribute." The Office Action cites Kamiya et al. at Page 1, first paragraph, as teaching this limitation. However, **neither there nor anywhere else in Kamiya et al. is such a step disclosed or suggested. The teaching that more than one user accesses the proxy is not a teaching of user replicas**, as used in this Application, at Page 22, beginning at line 14, and continuing through the subsequent pages (and as shown in Figs. 4A-4C). In particular, this Application refers to each user replica as "a local replica of the data collected by the server." Page 22, lines 20-21. Hence, Kamiya et al. fails to disclose or suggest any replica-related steps.

Steps (d) and (e) respectively recite, "submitting the update stipulated in step c to the replicas of selected users;" and "receiving at least one update from at least one user in the community and updating said user replica with the so received update." The Office Action cites Kamiya et al. at Pages 6 and 7 for such teachings. However, Kamiya et al. does not teach updating replicas in these sections; rather, Kamiya et al. discusses updating information among collections on the server.

Step (f) recites a step of "selectively displaying, through a user interface, at least one container." The Office Action relies on Figs. 4-6 of de Judicibus for a teaching of this limitation. However, de Judicibus does not disclose or suggest such displaying through a user interface in Figs. 4-6 (or anywhere else); rather, Figs. 4-6 show **conceptual** groupings of objects. This understanding of Figs. 4-6 is supported by the disclosure at Col. 3, lines 5-6 and Col. 4, lines 56-57.

For at least these reasons, it is respectfully submitted that Claim 15 is allowable over the cited prior art. Hence, Claims 1-3, 8, 11-14, 16-20, and 22-27, which depend from Claim 15, are also allowable.

Applicants further note, regarding the rejection of Claim 15, that in order to establish *prima facie* obviousness, the Examiner must show motivation to combine the prior art publications. MPEP § 2143.01; In re Lee 61 USPQ2d 1434 (Fed. Cir. 2002). In particular, the Federal Circuit warns against the misuse of "common knowledge" or "common sense" as a basis for combining references.

Bearing this in mind, De Judicibus patent describes a single-user GUI. **In no place does he mention or suggest sharing an object among a community of users**. In fact, there is no motivation to do this, because his invention is complete as is – a single-user GUI. He certainly does not suggest or mention, *inter alia*, updating replicas or propagating the updates, which are mandatory in a sharing community environment, but which are absolutely redundant in a "self-contained" graphic user interface, as in de Judicibus.

The fact that sharing information is generally known, *per se* (as discussed, e.g. , with respect to various prior art systems in the Background of the Invention section of this Application, including the cited Kamiya et al.), does not *ipso facto* provide to "one ordinary skilled in the art" a motivation to combine the "self-contained" de Judicibus publication with the Kamiya et al. publication.

Applicants further maintain that de Judicibus teaches away from the invention. In re Hedges, 783 F.2d 1038, 228 USPQ 685 (Fed. Cir. 1986) is a good example of the Federal Circuit's contribution to the jurisprudence on "teaching away." Hedges had stressed to the PTO that his invention incorporated the reaction to diphenyl sulfone at a temperature above its melting point of 127°C. Hedges argued before the board that the lower temperature shown by the prior art defeats any prima facie case of obviousness. The CAFC determined that the basic reference alone does not, as asserted by the PTO, support a *prima facie* case of obviousness. It was observed that the reference (Felix) makes clear that low temperatures are the desired conditions for the claimed reaction.

By analogy, de Judicibus indicated,

> in the state of the art operating system, this requires a manual intervention by the user or the creation for statically defined groups by a program. Anyway, the sub-groups so created should be maintained when the number of objects increases in order to

> avoid again having too many objects in a single group. Often this requires a rearrangement of all the groups and laborious maintenance operations.

(Column 2, lines 27-32). De Judicibus further indicates that "it is the object of the present invention to provide a technique which alleviates the above drawback" (col. 2, lines 34-35), i.e., *inter alia*, a technique that reduces the tedious manual maintenance from the user end.

In contrast, the claimed invention has different goals. The invention does not focus on whether the user maintains her local data manually or automatically. In fact, in certain embodiments, manual editing is permitted (see, e.g., page 22, lines 11 to 14, where it is explicitly stated that users may edit their trees at any time).

Accordingly, similar to In re Hedges, where prior art on low temperatures teaches away from high temperature, according to the invention, the prior art teaches away from a use, in certain embodiments of the invention, of manual editing.

Applicants, therefore, respectfully submit that a *prima facie* establishment of obviousness has not been met in respect of Claim 15 and that the prior art teaches away from the invention as defined in Claim 15.

Further regarding Claim 24, the Office Action asserts that de Judicibus, e.g., at Fig. 5, teaches a tree like that in Microsoft™ File Explorer. However, as amended, Claim 24 specifically refers to the tree being displayed "in a hierarchical, expandable and contractable fashion." Applicants respectfully submit that this is neither disclosed nor suggested in de Judicibus (or in Kamiya et al.). Hence, it is respectfully submitted that Claim 24 is allowable for this further reason.

Claim 27 recites the same limitations as Claim 13, discussed above. Therefore, the same

arguments applied to Claim 13 apply to Claim 27. For this further reason, it is respectfully submitted that Claim 27 is allowable over the cited prior art.

Claim 29 includes the limitations found in Claim 15 and is thus respectfully submitted to be allowable over the cited prior art for at least the same reasons for which Claim 15 is allowable over the cited prior art.

New Claim

New Claim 32 has been added. Claim 32 recites a machine-readable program storage device that embodies, in the form of a program, the steps found in Claims 15 and 29. This is supported in the disclosure at least at Pages 11-12 and in the claims as originally filed.

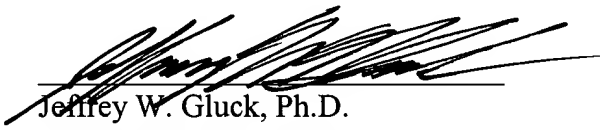
Conclusion

All of the stated grounds of objection and rejection have been properly traversed, accommodated, or rendered moot. Applicants therefore respectfully request that the Examiner reconsider all presently outstanding objections and rejections and that they be withdrawn. Applicants believe that a full and complete reply has been made to the outstanding Office Action and, as such, the present application is in condition for allowance. If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is hereby invited to telephone the undersigned at the number provided.

Prompt and favorable consideration of this Amendment is respectfully requested.

Respectfully submitted,

Date: December 16, 2002



Jeffrey W. Gluck, Ph.D.
Registration No. 44,457
VENABLE
P.O. Box 34385
Washington, D.C. 20043-9998
Telephone: (202) 962-4800
Direct Dial: (202) 216-8017
Telefax: (202) 962-8300

APPENDIX: MARKED-UP VERSIONS OF AMENDED PORTIONS OF THE APPLICATION

In the Specification:

The paragraph beginning at Page 6, line 1 has been replaced with the following rewritten paragraph:

Active Notebook (Torrance) allows users to label information with conceptual classifications and to organize[d] them into a taxonomy for later browsing and retrieval. The focus of Torrance's work is on clustering documents and identifying morphological concepts (keywords). Torrance does not deal with aspects of sharing such as replication, privacy and user interface.

The paragraph beginning at Page 8, line 11 has been replaced with the following rewritten paragraph:

There is, accordingly, a need in the art to provide a technique, which enables one to manage and possibly share objects among users, and which substantially overcomes the limitations of hitherto known techniques. There is a further [a] need in the art to provide a system that enables one to manage objects also in a single user environment.

The paragraph beginning at Page 11, line 21 has been replaced with the following rewritten paragraph:

computer readable program code for causing the computer to [providing] provide a set of containers, each associated with attributes from among said set;

The paragraph beginning at Page 11, line 27 has been replaced with the following rewritten paragraph:

computer readable program code for causing the computer to selectively display[ing], through a user interface, at least one container; an object is displayed in said container if a condition is met; the condition is applied to at least the following: at least one of the attributes of the container and at least one of the attributes of the object.

The three paragraphs beginning at Page 12, line 20 has been replaced with the following rewritten paragraphs:

- Privacy: users may easily define some URLs (and possibly attributes) as private. Privately marked URLs are encrypted in the server and in the replicas of that user, so that access is possible only using the key of the user. *Privacy* may be regarded as an attribute that is associated with selected folders. Bookmarks can be dynamically assigned to folders, and by the preferred embodiment they [inherent] inherit the attributes of the folder. Thus bookmarks that are assigned to *private* folders inherit the privacy attribute. Other attributes may be assigned to folders such as, for example, *History*, signifying that bookmarks associated to this folder have just been recently used. Another non-limiting example is *Hot* attribute associated to a folder that includes all those bookmarks which have just been recently added to folders.
- Simple, familiar view user interface: the implementation uses a familiar tree folder structure user interface, allowing the user to perform common operations with click and drop user interface. The approach resembles a known user interface, such as, for example, in the

MICROSOFT™ [folder explorer] Folder Explorer file system interface utility, and therefore reduces the time it takes to become familiar with the user interface. The *drag and drop* operation is utilized by one embodiment to assign attributes to bookmarks[)]. Having mapped the bookmark to a folder the attributes of the folder are automatically assigned to the bookmark.

- Display of bookmarks in folders through familiar user interface. As will be explained in greater detail below, assignment of attributes to bookmarks, e.g. by drag and dropping the specified bookmark to a folder, does not necessarily mean that the bookmark will be displayed in the specified folder. As will be elaborated in greater detail below, [In] in order for the bookmark to be displayed in the folder, it must meet a condition(s) that is (are) applied to the attributes of the container and the bookmark.

The paragraph beginning at Page 14, line 17 has been replaced with the following rewritten paragraph:

Fig. 5 illustrates an exemplary attributes dialog box, in accordance with [on] an embodiment of the invention;

The paragraph beginning at Page 15, line 6 has been replaced with the following rewritten paragraph:

A very popular interface for viewing bookmarks (as well as files, mail messages, etc.), is using a tree of folders, e.g. in accordance with the Microsoft™ [file explorer] File Explorer file system interface utility's user interface. In many applications, browsing the tree is a better way to

look for the right bookmark, rather than doing a textual search in the database. However, organizing shared bookmarks into folders is difficult. The same URL may be relevant to more than one folder; furthermore, the different users may prefer [a] different arrangements of folders (e.g., one user prefers top level folder "music" and sub-folder "shopping", whereas another user prefers the other way around, i.e. top level folder "shopping" and sub-folder "music"), or simply a different name for folders (e.g., "find" and "search").

The paragraph beginning at Page 16, line 21 has been replaced with the following rewritten paragraph:

The set of attributes is used to display bookmarks in, say, a tree of folders. A bookmarks displayed in a folder (or folders) if a condition is met. The condition is applied to at least the following: at least one of the attributes of the folder and at least one of the attributes of the bookmark. By a specific embodiment a bookmark is displayed in a folder if the bookmark and the folder share a common attribute.

The three paragraphs beginning at Page 17, line 4 have been replaced with the following rewritten paragraphs:

For a better understanding, attention is now directed to Figs. [1A-B] 1A-1B, illustrating a sample user interface represented as a tree of folder, in accordance with one embodiment of the invention.

Thus, the user interface 11 in Fig. 1A represented as a tree of folders (for, say, a first user in the community or a single user in a stand-alone environment) is similar to the known

Microsoft™ [file] File Explorer file system interface utility's tree representation. Each folder (e.g. "books" (12)) is associated with attributes and by this particular embodiment "shopping" and "book" (13). In the user interface of Fig. 1A, attributes are embraced by square brackets. Sample folder tree 14 in Fig. 1B represents the user interface for, say, a second user in the community.

The list of URLs (or bookmarks) that corresponds to a given folder[, are] is stored in the specified folder (not shown in Figs. 1A and 1B), similar to files that belong to a given folder in the Microsoft™ Windows Explorer file system interface utility.

The paragraph beginning at Page 20, line 2 has been replaced with the following rewritten paragraph:

After having assigned attributes to both the bookmarks and the folders, the bookmarks can be displayed. Bookmarks are displayed through a user interface, as illustrated e.g. in [Fig. 1] Figs. 1A-1B, and are organized e.g., into a tree of folders. Each folder has a name and [being] is assigned with one or more attributes (taken from the set of attributes). As described above, attributes are also assigned to each bookmark. By a preferred embodiment, if the folder contains a sub-folder, and the bookmark has also the attributes of the sub-folder, the bookmark is only displayed in the sub-folder (as a finer categorization). This logic is implemented by the following rule:

The paragraph beginning at Page 20, line 20 has been replaced with the following rewritten paragraph:

As specified above, a bookmark is displayed in a folder if a condition is met. By this specific example the condition includes a sub-condition that stipulates that an attribute (at least one) assigned to the bookmark is contained in the attributes of the folder in which the bookmark is displayed. The condition further includes a sub-condition [stipulated] stipulating that the bookmark does not appear in any folder contained in this folder.

The paragraph beginning at Page 21, line 3 has been replaced with the following rewritten paragraph:

As specified above, in accordance with the invention there is provided a user interface for enabling dynamic assignment of attributes to objects. In accordance with a preferred embodiment, using the same interface of [Fig. 1] Figs. 1A-1B, a drag and drop operation is used in order to insert/delete a bookmark. Thus, by this embodiment, upon dragging a URL into a folder, the URL inherits the attributes of that folder. The operation is simple and intuitive.

The paragraph beginning at Page 21, line 29 has been replaced with the following rewritten paragraph:

The consequence of a single drag and drop operation is further exemplified with reference to [Fig. 3] Figs. 3A-3D. Fig. 3A illustrates five user views (31 to 35). Having dragged and dropped *item 1* -- say, a bookmark (36 in Fig. 3B) to View 5, *item 1* is assigned with the attributes 1, 2, and 4. Consequently, *item 1* is automatically assigned to view 1 (31 - due to attributes 1 and 2), view 3 (33 - due to attribute 4) and view 4 (34 - due to attribute 1). The update among the views is implemented in a manner that is described in detail below.

The paragraph beginning at Page 27, line 12 has been replaced with the following rewritten paragraph:

"Global" attributes, which, as specified above, are shared among all the community users, are owned by the server (root), whereas "Local" and "Submit" attributes are owned by the corresponding client (user).

The paragraph beginning at Page 27, line 23 has been replaced with the following rewritten paragraph:

Turning now to the client, the user interface (shown in Figure 6)[,] is used to access bookmarks and to add bookmarks to the community's collection. Its appearance resembles the Microsoft™ Windows Explorer file system interface utility. The client also enables the user to perform tasks such as editing the tree, assigning attributes to folders or bookmarks, opening a bookmark in the active browser window, and so forth.

The paragraph beginning at Page 31, line 24 has been replaced with the following two rewritten paragraphs:

The invention has been described with a certain degree of particularity, but various alterations and modifications may be carried out, without departing from the scope of the following [Claims:] claims.

We claim:

In the Claims:

Claims 4-7, 9, 10, 20, 21, 28, 30, and 31 have been cancelled.

Claims 1, 2, 11, 12, 15, 19, 24, and 29 have been amended as follows:

1. (Amended) A method according to claim 15, further adapted to [for] manage [managing] objects for at least one user, comprising:

[(a)] (1) providing a set of attributes;

[(b)] (2) providing a set of containers, each associated with attributes from among said set of attributes;

[(c)] (3) providing a user interface for dynamically assigning attributes to said objects;

[(d)] (4) selectively displaying, through [a] the user interface, at least one container[;], wherein an object is displayed in said container if a condition applied to at least one of the attributes of the container and at least one of the attributes of the object is met[; the condition is applied to at least the following: at least one of the attributes of the container and at least one of the attribute of the object].

2. (Amended) The method according to Claim 1, wherein said assigning, stipulated in step [(c)] (3), includes mapping an object from among said objects to at least one container from among said containers and inheriting attributes [thereof] of said container.

11. (Amended) The method according to claim [4]15, further comprising the steps of:

categorizing selected objects as private; and encrypting said objects with a user unique key.

12. (Amended) The method according to claim [4] 15, wherein selected containers in at least one of said user replicas are assigned each with a private attribute; and further comprising the step of encrypting any object so assigned to a container from among said containers [is encrypted] using a unique key.

15. (Amended) A method for sharing objects among a community of users[;], wherein each user is associated with a respective set of attributes such that at least one attribute is common to at least two of said users[;], the method comprising executing the following steps for each user in the community:

(a) providing a user replica that includes objects that are assigned, each, with at least one attribute;

(b) providing a set of containers associated, each, with attributes from among said set of attributes;

(c) providing a user interface for generating an update in said replica;

(d) submitting the update stipulated in step (c) to the replicas of selected users;

(e) receiving at least one update from at least one user in the community and [update] updating said user replica with the so received update; and

(f) selectively displaying, through [a] the user interface, at least one container[;], wherein an object is displayed in said container if a condition applied to at least one of the attributes of the container and at least one of the attributes of the object is met[; the condition is applied to at least the following: at least one of the attributes of the container and at least one of the attribute of the object].

19. (Amended) The method according to Claim 16, wherein said update includes updating at least one attribute in said set of attributes; said updating of attribute is selected from the group that includes: adding an attribute, deleting an attribute, editing an attribute, change status of attribute from Global to Local or [*vice versa]* Local to Global, joining two or more attributes into a single attribute.

24. (Amended) The method according to Claim[s] 22, wherein said tree [being Microsoft™ File Explorer like] is displayed in a hierarchical, expandable and contractable manner, and wherein said containers [being] comprise folders.

29. (Amended) A system for sharing objects among a community of users[;], wherein the system includes at least one server communicating through a network with users, each being associated with a processor and associated memory and display that includes a respective set of attributes such that at least one attribute is common to at least two of said users[;], and the respective processor and associated memory and display are configured to [executing] execute the following steps:

(a) providing to each user a user replica that includes objects that are assigned, each, with at least one attribute;

(b) providing a set of containers associated, each, with attributes from among said set of attributes;

(c) providing a user interface for generating an update in said replica;

(d) submitting through said at least one server the update stipulated in step (c) to the replicas of selected users;

(e) receiving through said at least one server at least one update from at least one user in the community and [update] updating said user replica with the so received update; and

(f) selectively displaying, through [a] the user interface, at least one container[;], wherein an object is displayed in said container if a condition applied to at least one of the attributes of the container and at least one of the attributes of the object is met[; the condition is applied to at least the following: at least one of the attributes of the container and at least one of the attribute of the object].

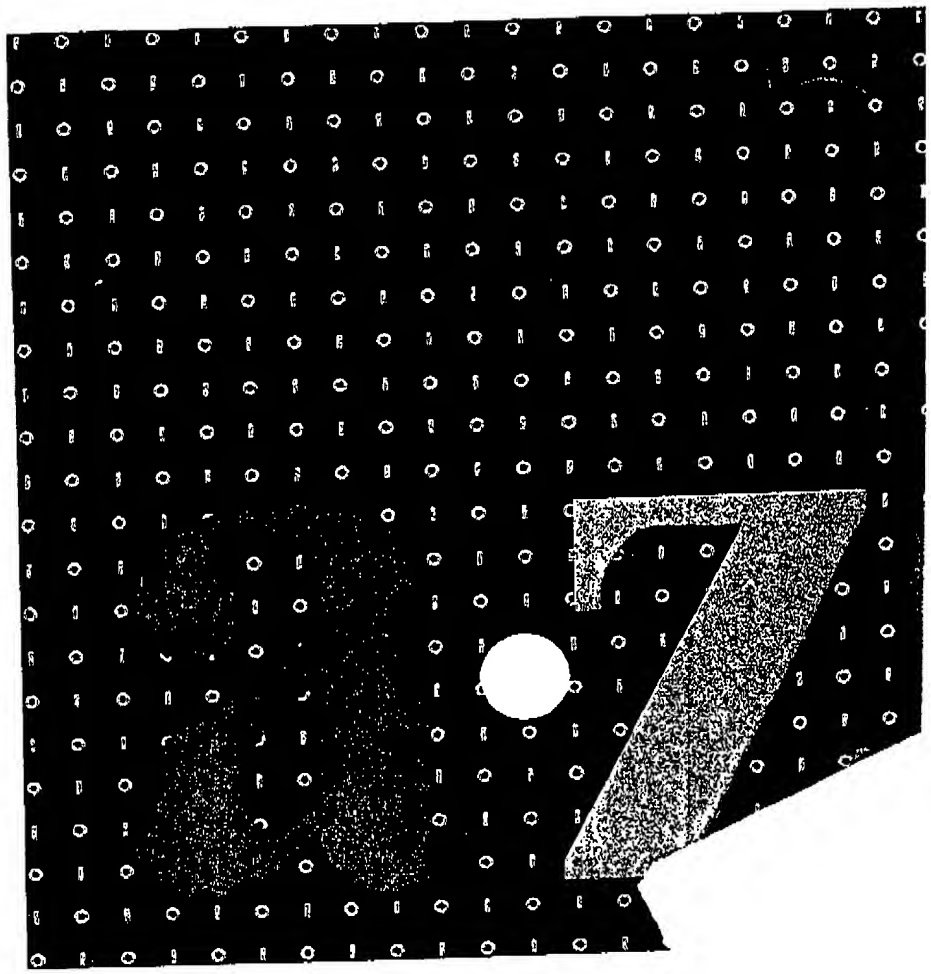
New Claim 32 has been added. The text of new Claim 32 may be found in the body of the Amendment and Reply.

::ODMA\PCDOCS\DC2DOCS1\421355\1
VBHC Rev. 12/16/02 rpa

Dictionary of Computing

SC20-16

Information Processing, Personal Computing, Telecommunications, Office Systems, IBM-specific Terms



SC20-1699-8

IBM

Dictionary of Computing

Information Processing, Personal Computing, Telecommunications, Office Systems, IBM-specific Terms

PROPERTY OF
DR. REINHOLD COHN & PARTNERS
9-11 Yavne Street, Tel-Aviv
Date received : 12/93
Class: No: :
Acc: No: : 23001

unprotected field. A displayed field in which a user can enter, modify, or delete data. Contrast with protected field.

unprotected storage. (1) In the AS/400* system, the part of the system auxiliary storage pool that is not protected by the checksum option. (2) In the AS/400 system, the storage reserved for temporary objects and internal machine data while a job is running.

unpunched litho master. In a duplicator, a litho master with unperforated edges. (T)

unqualified call. In IMS/VS, a DL/I call that does not contain a segment search argument.

unqualified segment search argument. In IMS/VS, a segment search argument (SSA) that contains only a segment name specifying the segment type to be accessed.

unrecoverable abend. An error condition that results in abnormal termination of a program. Contrast with recoverable abend.

unrecoverable error. Synonym for irrecoverable error.

unrecoverable transaction. In IMS/VS, an inquiry transaction that is not recovered in the event of a failure.

unsolicited interrupt. In the AIX* operating system, an interrupt that is sent to a virtual machine when its last virtual terminal is closed. See interrupt.

unsolicited message. (1) In MSS, a message from the mass storage control to the primary processing unit that is not requested or expected by the processing unit. (2) A message from the ACF/VTAM* program to a program operator that is unrelated to any command entered by the program operator. Contrast with solicited message.

unstable state. (1) In a circuit, a state in which the circuit remains for a finite period of time at the end of which it returns to a stable state without the application of a pulse. (T) (2) Synonymous with metastable state, quasistable state.

unstratified language. (1) A language that can be used as its own metalanguage; for example, most natural languages. (I) (A) (2) Contrast with stratified language.

unsuccessful execution. In COBOL, the attempted execution of a statement that does not result in the execution of all the operations specified by that statement. The unsuccessful execution of a statement does not affect any data referenced by that statement, but may affect status indicators.

unviewable. In AIX* graphics, pertaining to a mapped window with an unmapped ancestor. Contrast with viewable.

unwind. To state explicitly and in full, without the use of modifiers, all the instructions involved in the execution of a loop. (I) (A)

UP. Unnumbered poll (SDLC).

UPC. Universal product code.

update. (1) To add, change, or delete items. (2) To modify a master file with current information according to a specified procedure.

update authority. (1) The ability to add, change, or cancel items. (2) In the AS/400* system, a data authority that allows the user to change the data in an object, such as a journal, a message queue, or a data area. See also add authority, delete authority, read authority.

updated-record mark. In the IBM* 3790 Communication System, an indication, set in the control information for an index record, that the record has been changed. See also mark function.

update file. A file from which a program reads a record, updates fields in the record, and writes the record back into the location from which it came.

update install. The process of modifying an 8100/DPCX system. Contrast with full install.

update intent. In IMS/VS, the scheduling intent type that permits application programs to be scheduled with any number of other programs except those with exclusive intent.

update mark. In DPCX, a byte in the control record for an indexed record that can be written by a user program to indicate access or change to the record.

update-only recovery. An IMS/VS facility that allows the user to define inquiry transactions as unrecoverable.

update operation. An I/O operation that modifies the information in a file.

update rights. The authority to change the entries in an object. Contrast with add rights, delete rights, read rights.

update script. In the AIX* operating system, a shell procedure or executable file created by the developer of an application program to update a program. The script file must follow specific guidelines in order to be compatible with the program update tools that are provided in the operating system.

update transaction. In IMS/VS, a transaction in a system with the DC feature with capabilities for updating a database. Update transactions are recoverable.

upgrade set. In VSAM, all of the alternate indexes that are to be updated whenever there is a change to the data component of the related base cluster.

upline. Pertaining to the direction opposite to the direction of transmission. Contrast with downline.

uplink. Pertaining to data transmission from a data station to the headend. (T) Contrast with downlink.

upload. (1) To transfer programs or data from a connected device, typically a personal computer, to a computer with

DEC 16 2002
PATENT & TRADEMARK OFFICE

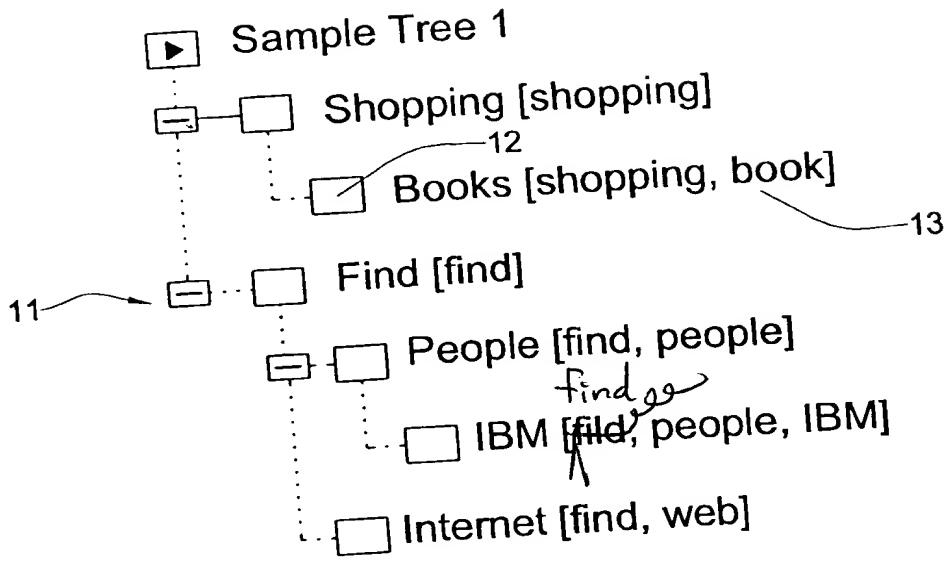


FIG. 1A

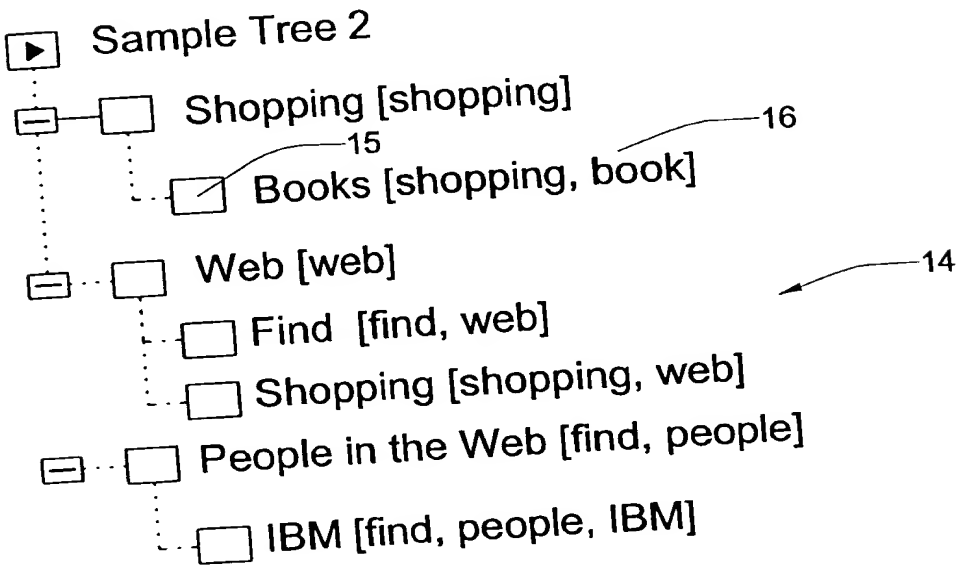


FIG. 1B

OIPE JC76
DEC 16 2002
PATENT & TRADEMARK OFFICE

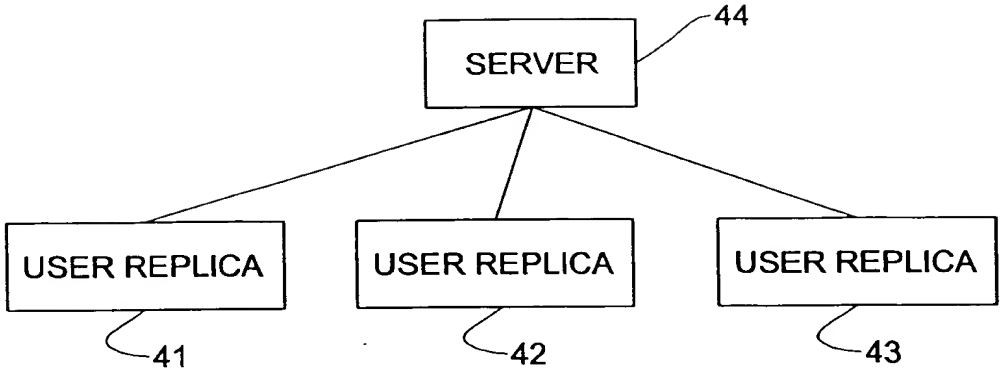


FIG. 4A

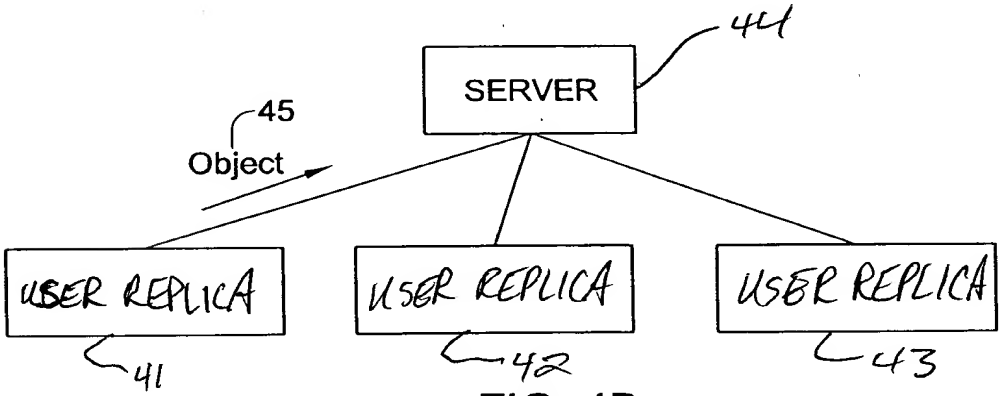


FIG. 4B

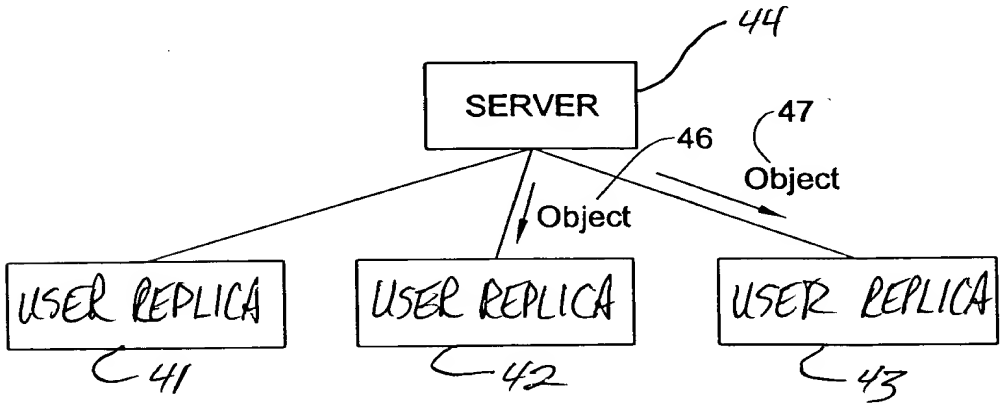


FIG. 4C

PATENT & TRADEMARK OFFICE
DEC 16 2002

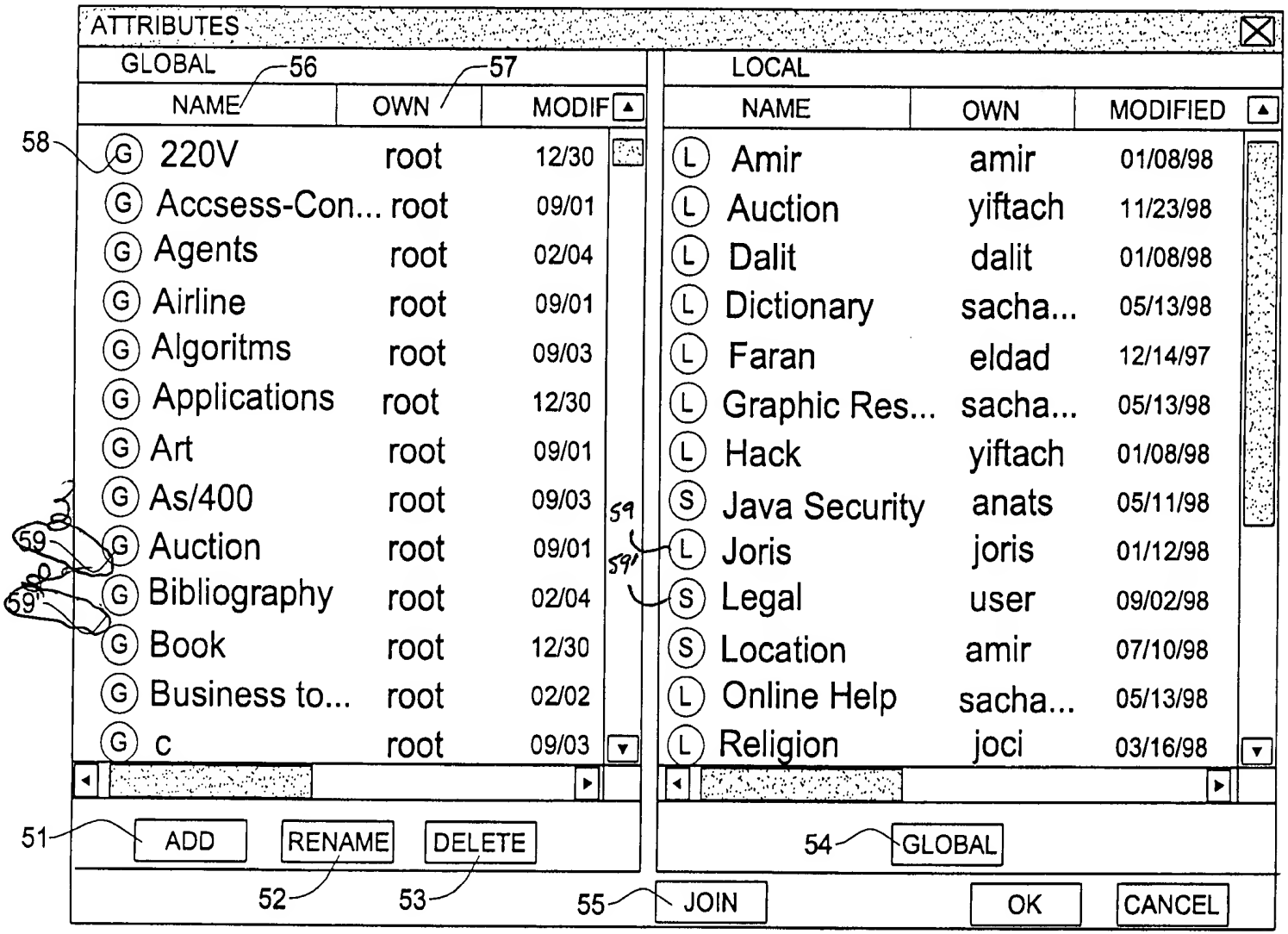


FIG. 5